Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claim in the application:

Listing of Claims:

Claim 1 (currently amended) A data dictionary comprising:

an inverse fault-tolerant decoder implemented for an error-correction code configured to transform a data vector into a plurality of predetermined index values~~, each of said index values specifying a location within a first address space~~;

~~combinational~~ logic configured to combine pairs of said index values by concatenating said pairs of said index values to form corresponding pairwise combined hash indices~~, specifying a location within a second address space, said second address space being greater than said first address space~~; and

data storage configured as a hash table~~ addressable throughout said second address space, said hash table~~ referencing indexed data stored in the data dictionary corresponding to said pairwise combined hash indices.

Claim 2 (original) The data dictionary according to claim 1 wherein said data vector comprises a bit-attribute vector.

Claim 3 (original) The data dictionary according to claim 1 wherein said inverse fault-tolerant decoder implements a reverse perfect error correction code.

Claim 4 (original) The data dictionary according to claim 3 wherein said reverse perfect error correction code comprises a reverse Golay code.

Claim 5 (original) The data dictionary according to claim 1 wherein said inverse fault tolerant decoder is further configured to identify said data vector as one of (i) a border vector type located at a border of a decoding sphere and (ii) a non-border vector type located interior to said decoding sphere.

Claim 6 (currently amended) ~~A~~ The data dictionary according to claim 1 ~~comprising:~~

~~an inverse fault-tolerant decoder implemented for an error-correction code configured to transform a data vector into a plurality of predetermined index values;~~

~~combinational logic configured to combine pairs of said index values to form corresponding pairwise combined hash indices; and~~

~~data storage configured as a hash table referencing indexed data corresponding to said pairwise combined hash indices,~~

wherein said inverse fault-tolerant decoder is configured to:

identify said data vector as a border vector type;

define an offset of said data vector from a center of a decoding sphere of an error-correction code implemented by said inverse fault-tolerant decoder; and

identify ~~all~~ ~~possible~~ offsets from adjacent decoding spheres of said error-correction code ~~until said combinations fill in all bit positions corresponding to said data vector such that centers of said adjacent decoding spheres correspond to said index values~~.

Claim 7 (currently amended) ~~A~~ The data dictionary according to claim 1 ~~comprising:~~

~~an inverse fault-tolerant decoder implemented for an error-correction code configured to transform a data vector into a plurality of predetermined index values;~~

~~combinational logic configured to combine pairs of said index values to form corresponding pairwise combined hash indices; and~~

~~data storage configured as a hash table referencing indexed data corresponding to said pairwise combined hash indices,~~

wherein said fault-tolerant decoder implements a reverse Golay code and is configured to:

identify said data vector as a non-border vector type;

identify an offset vector of said data vector from a center of a central index decoding sphere representing a specified offset distance;

identify centers of adjacent decoding spheres within said specified offset distance of said data vector; and

~~combines~~ concatenate (i) hash indexes corresponding to said centers of each of said adjacent decoding spheres with (ii) a hash index corresponding to said center of said central index decoding sphere to form respective pairs of hash indexes.

Claim 8 (currently amended) A method of accessing a dictionary comprising the steps of:

transforming a data vector into a plurality of predetermined index values~~, each of said index values specifying a location within a first address space~~;

~~combining~~ concatenating pairs of said index values to form corresponding pairwise combined hash indices~~, each of said pairwise combined hash indices specifying a location within a second address space, said second address space being greater than said first address space~~; and

referencing indexed data stored in ~~a hash table throughout said second address space~~ the dictionary corresponding to said pairwise combined hash indices.

Claim 9 (original) The method according to claim 8 wherein said data vector comprises a bit-attribute vector.

Claim 10 (original) The method according to claim 8 wherein said transforming step implements a reverse perfect error correction code.

Claim 11 (original) The method according to claim 10 wherein said reverse perfect error correction code comprises a reverse Golay code.

Claim 12 (original) The method according to claim 8 wherein said transforming step further includes a step of identifying said data vector as one of (i) a border vector type located at a border of a decoding sphere and (ii) a non-border vector type located interior to said decoding sphere.

Claim 13 (currently amended) [[A]] The method of accessing a dictionary according to claim 8 ~~comprising the steps of:~~

~~transforming a data vector into a plurality of predetermined index values;~~

~~combining pairs of said index values to form corresponding combined hash indices; and~~

~~referencing indexed data stored in a hash table corresponding to said combined hash indices,~~

wherein said transforming step further includes the steps of

(i) identifying said data vector as a border vector type,

(ii) defining an offset of said data vector from a center of a decoding sphere of an error-correction code implemented by said inverse fault-tolerant decoder, and

(iii) identifying ~~all possible~~ offsets from adjacent decoding spheres of said error-correction code ~~until said combinations fill in all bit positions corresponding to said data vector such that centers of said adjacent decoding spheres correspond to said index values~~.

Claim 14 (currently amended) [[A]] The method of accessing a dictionary according to claim 8 ~~comprising the steps of:~~

~~transforming a data vector into a plurality of predetermined index values;~~

~~combining pairs of said index values to form corresponding combined hash indices; and~~

~~referencing indexed data stored in a hash table corresponding to said combined hash~~

~~indices,~~

wherein said transforming step further comprises the steps of

(i) identifying said data vector as a non-border vector type,

(ii) identifying an offset vector of said data vector from a center of a central index decoding sphere representing a specified offset distance,

(iii) identifying centers of adjacent decoding spheres within said specified offset distance of said data vector, and

(iv) ~~combining~~ concatenating (i) hash indexes corresponding to said centers of each of said adjacent decoding spheres with (ii) a hash index corresponding to said center of said central index decoding sphere to form respective pairs of hash indexes.

Claim 15 (currently amended) A data dictionary stored on a computer readable media, said data dictionary comprising:

inverse fault-tolerant decoder logic configured to transform a data vector into a plurality of predetermined index values~~, each of said index values specifying a location within a first address space~~;

~~combinational~~ logic configured to ~~combine~~ concatenate pairs of said index values to form corresponding pairwise combined hash indices~~, each of said pairwise combined hash indices specifying a location within a second address space, said second address space being greater than said first address space~~; and

a data storage structure configured as a hash table ~~addressable throughout said second address space, said hash table~~ referencing indexed data of said data dictionary and corresponding to said pairwise combined hash indices.

Claim 16 (original) The data dictionary according to claim 15 wherein said data vector comprises a bit-attribute vector.

Claim 17 (original) The data dictionary according to claim 15 wherein said inverse fault-tolerant decoder implements a reverse Golay code.

Claim 18 (original) The data dictionary according to claim 15 wherein said inverse fault tolerant decoder logic is further configured to identify said data vector as one of (i) a border vector type located at a border of a decoding sphere and (ii) a non-border vector type located interior to said decoding sphere.

Claim 19 (currently amended) [[A]] The data dictionary according to claim 15 ~~stored on a computer readable media, said data dictionary comprising:~~

~~inverse fault-tolerant decoder logic configured to transform a data vector into a plurality of predetermined index values;~~

~~combinational logic configured to combine pairs of said index values to form corresponding pairwise combined hash indices; and~~

~~a data storage structure configured as a hash table referencing indexed data corresponding to said pairwise combined hash indices,~~

wherein said inverse fault-tolerant decoder logic is configured to:

identify said data vector as a border vector type[[,]] ;

define an offset of said data vector from a center of a decoding sphere of an error-correction code implemented by said inverse fault-tolerant decoder; and

identify ~~all~~ ~~possible~~ offsets from adjacent decoding spheres of said error-correction code ~~until said combinations fill in all bit positions corresponding to said data vector such that centers of said adjacent decoding spheres correspond to said index values~~.

Claim 20 (currently amended) [[A]] The data dictionary according to claim 15 ~~stored on a computer readable media, said data dictionary comprising:~~

~~inverse fault-tolerant decoder logic configured to transform a data vector into a plurality of predetermined index values;~~

~~combinational logic configured to combine pairs of said index values to form corresponding pairwise combined hash indices; and~~

~~a data storage structure configured as a hash table referencing indexed data corresponding to said pairwise combined hash indices,~~

wherein said fault-tolerant decoder logic implements a reverse Golay code and is configured to:

identify said data vector as a non-border vector type;

identify an offset vector of said data vector from a center of a central index decoding sphere representing a specified offset distance;

identify centers of adjacent decoding spheres within said specified offset distance of said data vector; and

~~combines~~ concatenate (i) hash indexes corresponding to each of said centers of said adjacent decoding spheres with (ii) a hash index corresponding to said center of said central index decoding sphere to form respective pairs of hash indexes.

Claim 21 (currently amended) The data dictionary according to claim 1 wherein said ~~combinational~~ logic is configured to combine pairs of said index values by pairing said index values in lexicographical order to form said corresponding pairwise combined hash indices.

Claim 22 (cancelled)

Claim 23 (previously presented) The method according to claim 8 wherein said step of combining includes combining said pairs of said indices values in lexicographical order to form said corresponding pairwise combined hash indices.

Claim 24 (cancelled)

Claim 25 (currently amended) The data dictionary according to claim 15 wherein said ~~combinational~~ logic is configured to combine pairs of said index values in lexicographical order to form said corresponding pairwise combined hash indices.

Claim 26 (cancelled)

Claim 27 (new) A method of storing information in a memory comprising the steps of:

identifying a plurality of distortions of a data vector;

hashing said data vector and said distortions of said data vector to form respective hash indices;

concatenating pairs of said hash indices; and

storing said data vector in the memory at addresses referenced by each of said pairs of hash indices.

Claim 28 (new) A method of searching a memory comprising the steps of:

identifying a plurality of distortions of a target data vector;

hashing said target data vector and said distortions of said target data vector to form respective hash indices;

concatenating pairs of said hash indices; and

comparing said target data vector with vectors stored in the memory at addresses referenced by each of said pairs of hash indices.

Claim 29 (new) A data dictionary comprising:

logic configured to identify a plurality of distortions of a data vector;

a decoder configured to hash said data vector and said distortions of said data vector to form respective hash indices;

logic configured to concatenate pairs of said hash indices; and

data storage configured to store said data vector in the data dictionary at addresses referenced by each of said pairs of hash indices.